

# Network encoders with Transformers

GraphBERT and related models

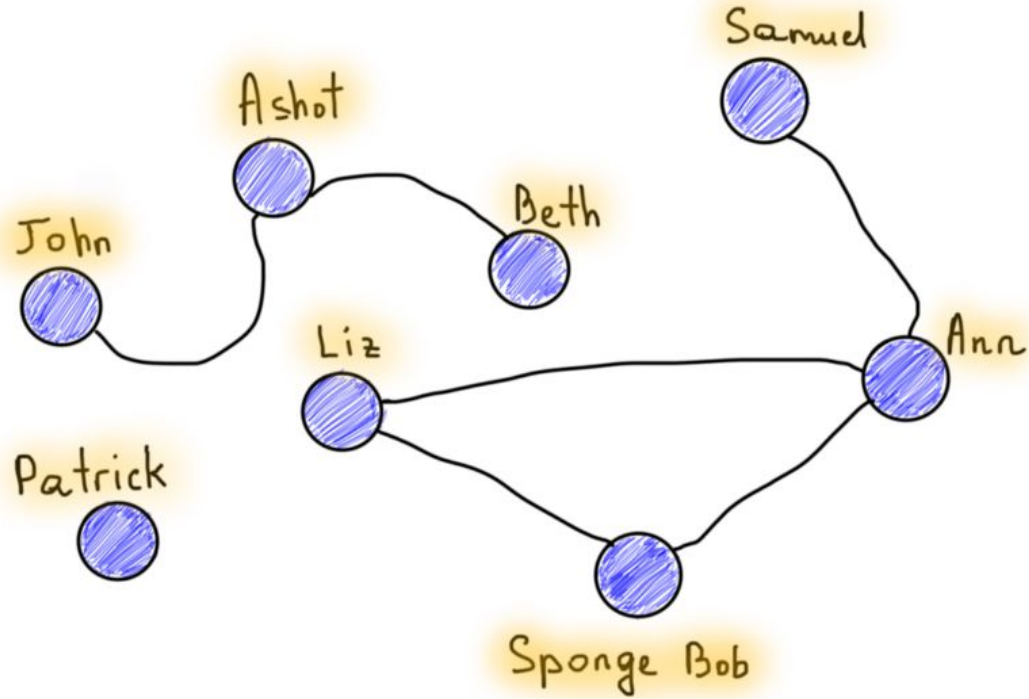
Irina Nikishina, PhD  
Universität Hamburg

*some slides are borrowed from  
WWW-18 Tutorial  
“Representation Learning on Networks”  
presented by Yure Leskovec*

# Course outline

1. Motivation for Transformer. Attention. The original Transformer architecture.
2. Transformer-based Encoders. Masked language models based on the Transformer architecture. BERT and related models.
3. Classification and sequence tagging with Transformers. Using encoders to generate feature representation for various NLU tasks.
4. Transformer-based Decoders. Generation of text based on the Transformer architecture. GPT and related decoders. Text generation methods. Prompt tuning.
5. Prompt and Instruction tuning. Reinforcement Learning from Human Feedback (RLHF), ChatGPT, and related models.
6. Sequence to sequence tasks: machine translation, text detoxification, question answering, dialogue. Technical tricks for training and inference: infrastructure and performance.
7. Multilingual language models based on the Transformer architecture.
8. Efficient Transformers.
9. Compression of transformer models.
10. Network encoders with Transformers.
11. Multimodal and vision Transformers.
12. Transformers for tabular data.
13. Transformers for event sequences.

# Graphs: recap





# Схема метро Москвы 2023



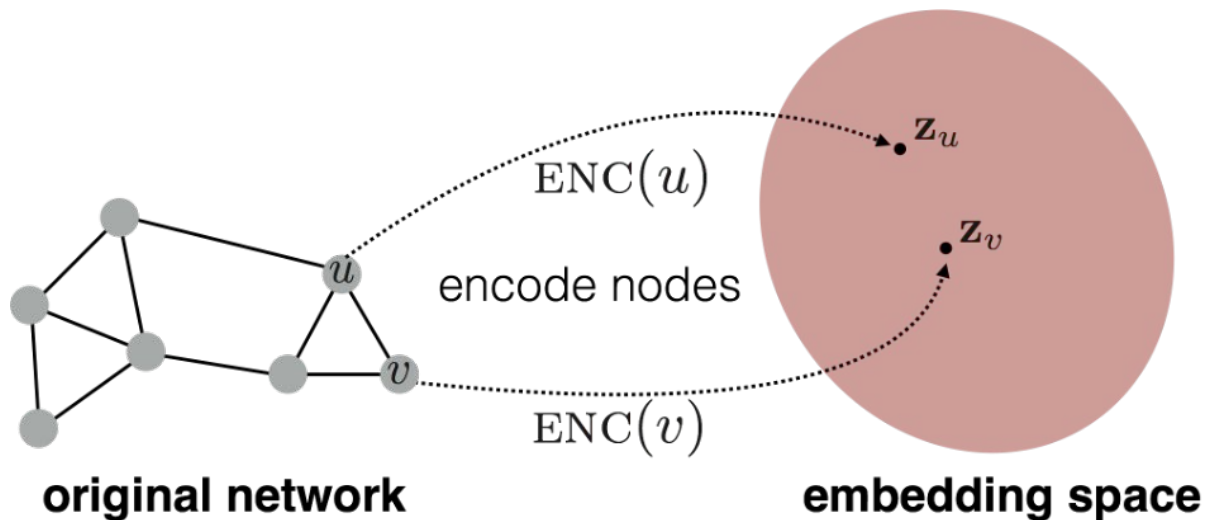
светло-серым цветом выделены станции, которые ремонтируются

- Сокольническая
- Замоскворецкая
- Арбатско-Покровская
- Филевская
- Кольцевая
- Калужско-Рижская
- Таганско-Краснопресненская
- Калининско-Солнцевская
- Серпуховско-Тимирязевская
- Люблинско-Дмитровская
- Бутовская (ленточное метро)
- Серпуховско-Тимирязевская
- Московское центральное кольцо (МЦК)
- Большая кольцевая линия (БКЛ)
- Бутовская (ленточное метро)
- Монокольцовая транспортная система

- 1 Красная площадь и Кремль
- 2 Храм Христа Спасителя
- 3 Поклонная гора
- 4 Третьяковская галерея
- 5 Останкинская башня
- 6 Усадьба Царицыно
- 7 Усадьба Коломенское
- 8 Зоопарк
- 9 Университет и Воробьевы горы
- 10 ВДНХ (ВДЦ)
- 11 Москва-Сити и Экспоцентр
- 12 Крокус Экспо

# Node embeddings

Goal is to encode nodes so that **similarity in the embedding space** (e.g., **dot product**) approximates **similarity in the original network**.



# Graph Neural Networks

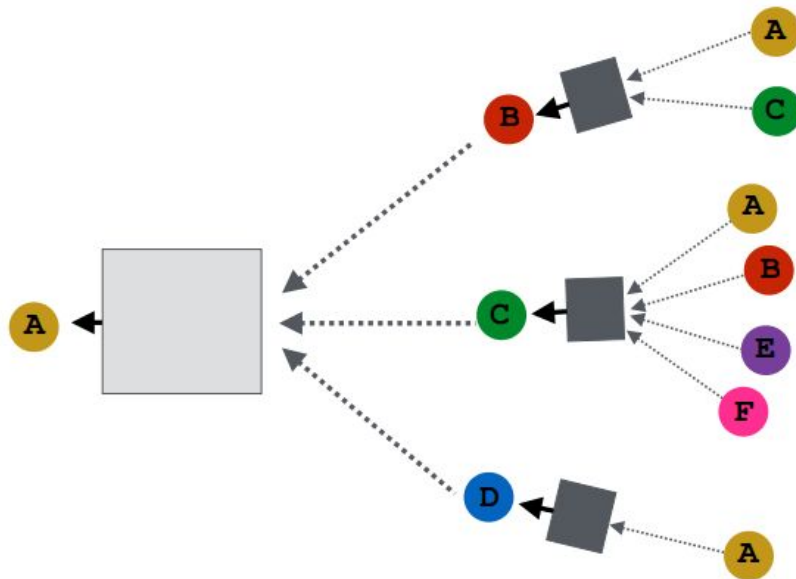
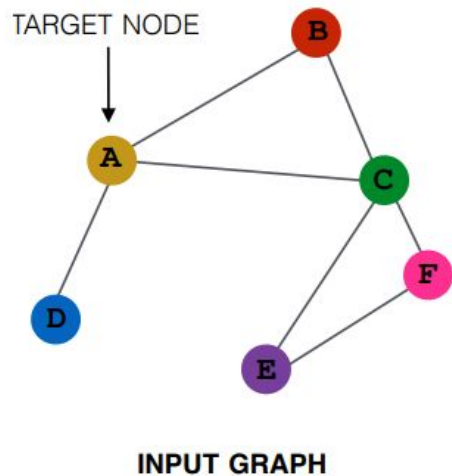
Assume we have a graph  $G$ :

- $V$  is the vertex set.
- $A$  is the adjacency matrix (assume binary).

$X \in \mathbb{R}^{m \times |V|}$  is a matrix of node features.

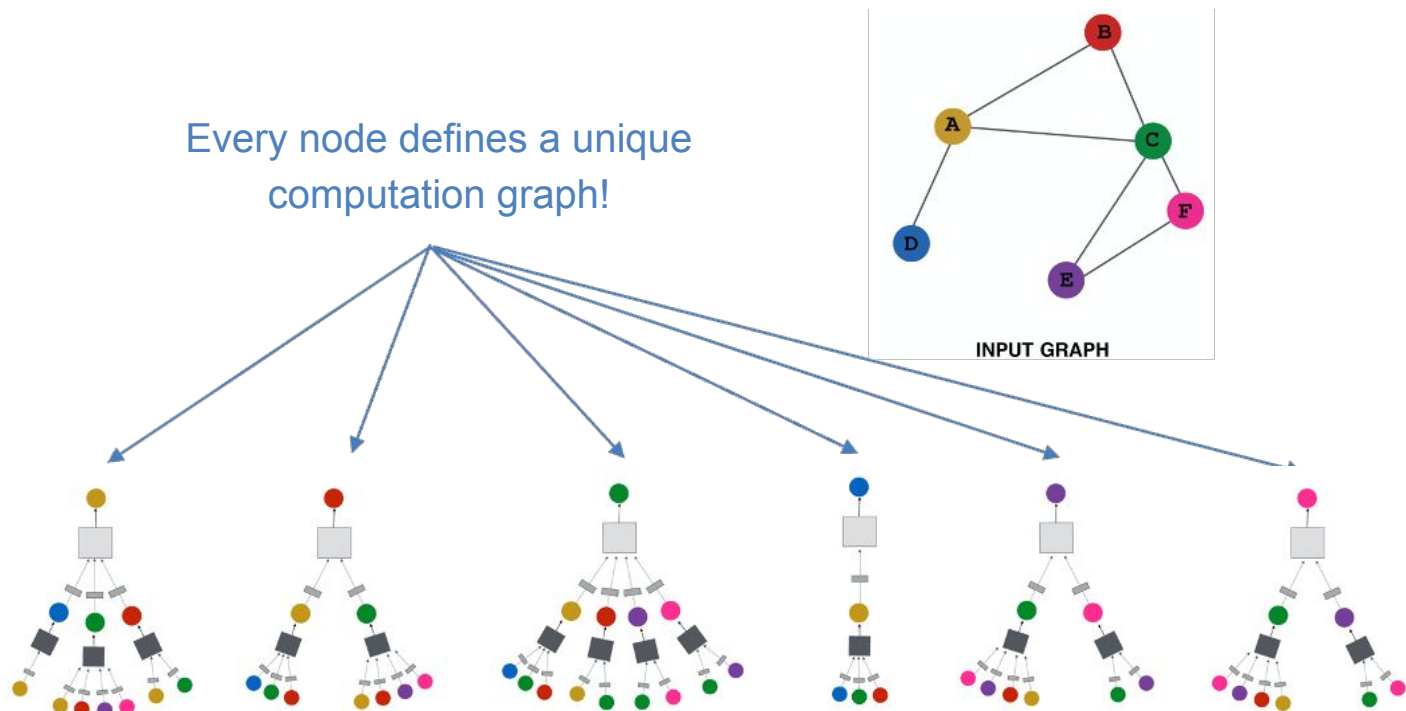
- Categorical attributes, text, image data
- E.g., profile information in a social network.
- Node degrees, clustering coefficients, etc.
- Indicator vectors (i.e., one-hot encoding of each node)

# Graph Convolutional Networks



T.N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, *arXiv preprint* 31  
*arXiv:1609.02907* (2016).

# Graph Convolutional Networks



T.N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, *arXiv preprint* 31

*arXiv:1609.02907* (2016). <https://arxiv.org/pdf/1609.02907.pdf>



# Graph Convolutional Networks

Trainable weight matrices  
(i.e., what we learn)

$$\mathbf{h}_v^0 = \mathbf{x}_v$$
$$\mathbf{h}_v^k = \sigma \left( \mathbf{W}_k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|} + \mathbf{B}_k \mathbf{h}_v^{k-1} \right), \quad \forall k \in \{1, \dots, K\}$$
$$\mathbf{z}_v = \mathbf{h}_v^K$$

# Graph Attention Networks (GAT)

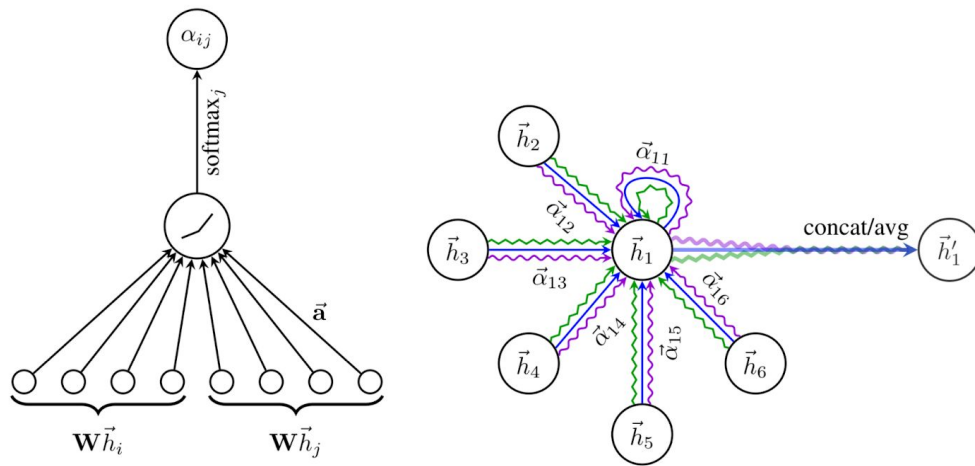
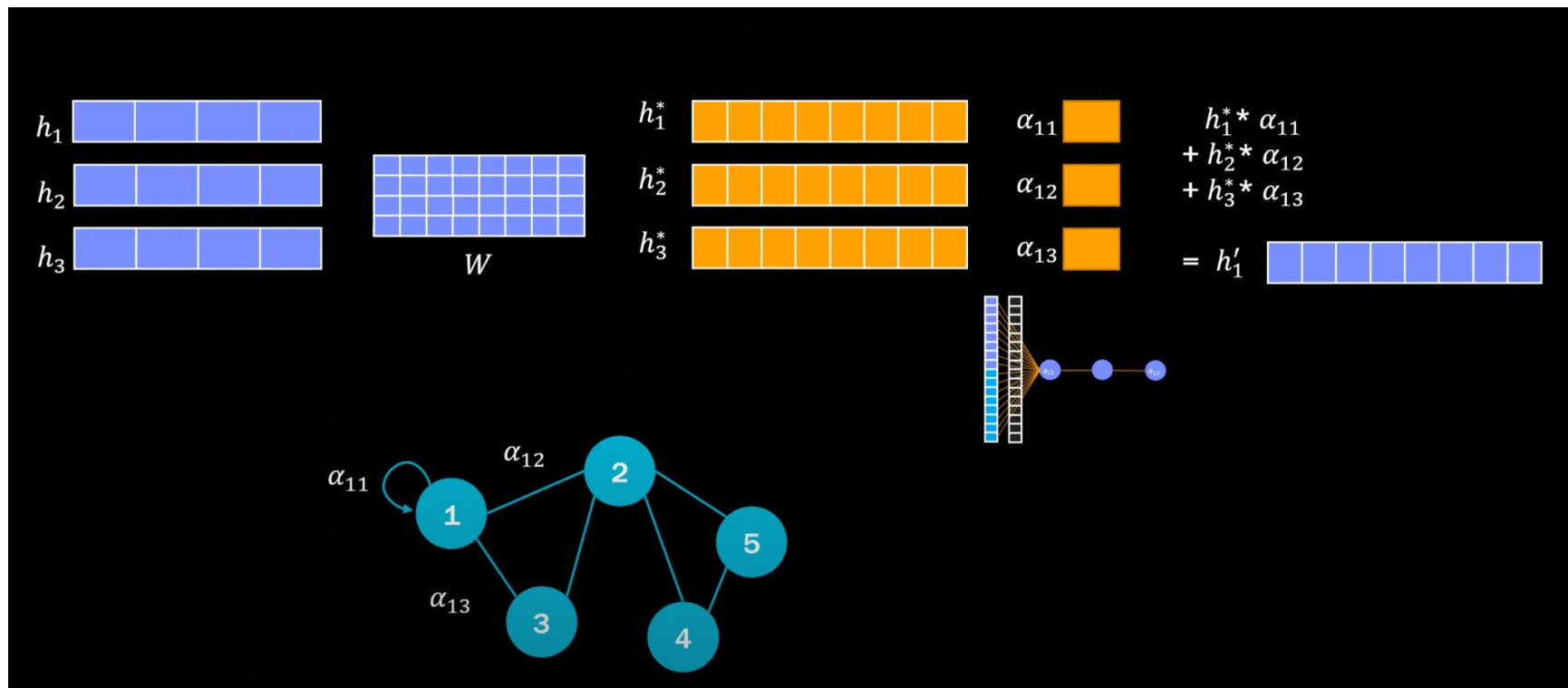


Figure 1: **Left:** The attention mechanism  $a(W\vec{h}_i, W\vec{h}_j)$  employed by our model, parametrized by a weight vector  $\vec{a} \in \mathbb{R}^{2F'}$ , applying a LeakyReLU activation. **Right:** An illustration of multi-head attention (with  $K = 3$  heads) by node 1 on its neighborhood. Different arrow styles and colors denote independent attention computations. The aggregated features from each head are concatenated or averaged to obtain  $\vec{h}'_1$ .

# Graph Attention Networks (GAT)



# Graph-BERT

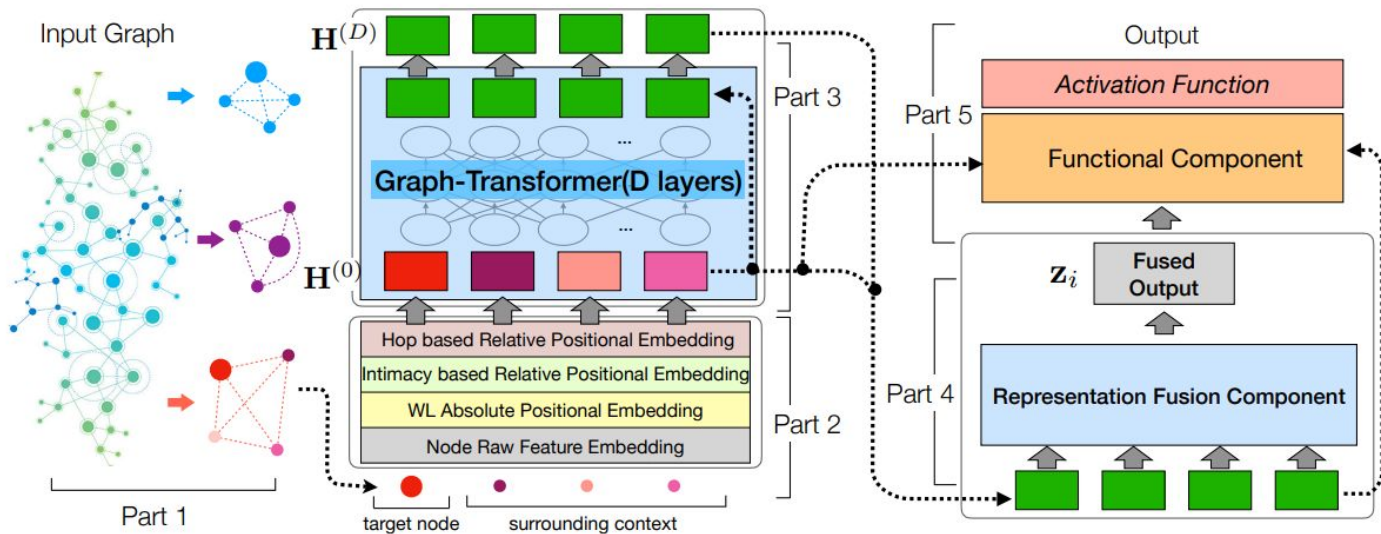
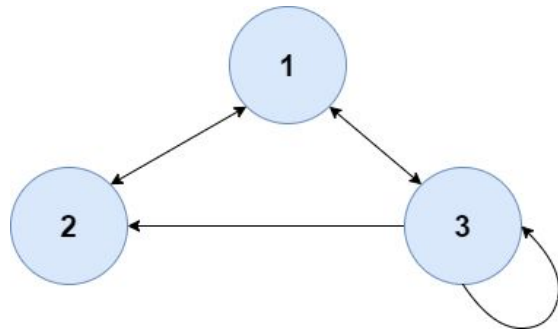


Figure 1: Architecture of the GRAPH-BERT Model. (Part 1: linkless subgraph batching; Part 2: node input vector embeddings; Part 3: graph transformer based encoder; Part 4: representation fusion; Part 5: functional component. Depending on the target application task, the function component will generate different output. In the sampled subgraphs, it covers both the target node and the surrounding context nodes.)

# Subgraph sampling

$$\mathbf{S} = \alpha \cdot (\mathbf{I} - (1 - \alpha) \cdot \bar{\mathbf{A}})^{-1}$$

$$\bar{\mathbf{A}} = \mathbf{A}\mathbf{D}^{-1}$$



$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

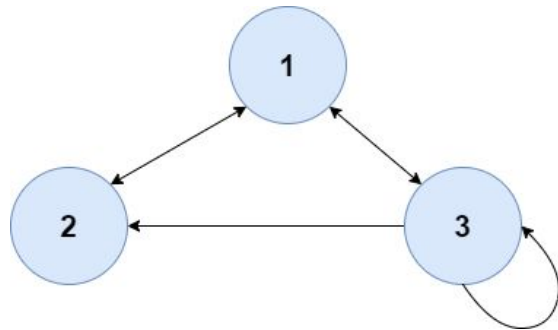
$$\mathbf{D} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

$$\bar{\mathbf{A}} = \begin{bmatrix} 0 & 1 & \frac{1}{3} \\ 0.5 & 0 & \frac{1}{3} \\ 0.5 & 0 & \frac{1}{3} \end{bmatrix}$$

# Subgraph sampling

$$\mathbf{S} = \alpha \cdot (\mathbf{I} - (1 - \alpha) \cdot \bar{\mathbf{A}})^{-1}$$

$$\bar{\mathbf{A}} = \mathbf{A} \mathbf{D}^{-1}$$



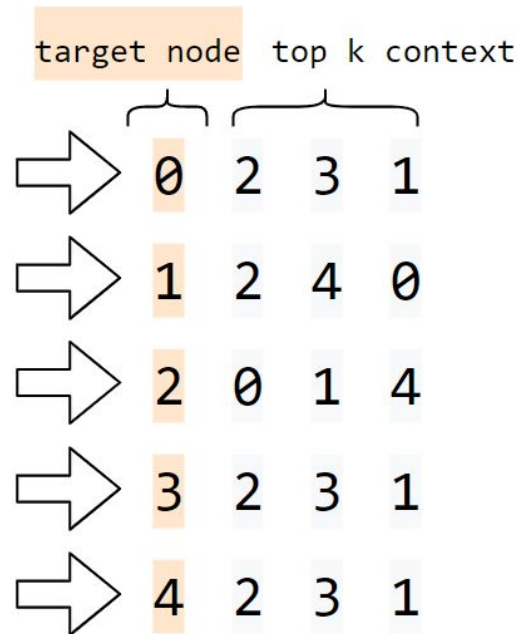
$$\mathbf{S} = (1 - \alpha) \cdot \mathbf{A} + \alpha \cdot \frac{1}{n} \mathbf{I}$$

$$\mathbf{S} = (1 - 0.15) \cdot \begin{bmatrix} 0 & 1 & \frac{1}{3} \\ 0.5 & 0 & \frac{1}{3} \\ 0.5 & 0 & \frac{1}{3} \end{bmatrix} + 0.15 \cdot \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix}$$

$$\mathbf{S} = \begin{bmatrix} 0.04995 & 0.89995 & 0.333 \\ 0.47495 & 0.04995 & 0.333 \\ 0.47495 & 0.04995 & 0.333 \end{bmatrix}$$

# Subgraph sampling

	0	1	2	3	4
0	0.014	0.209	0.231	0.228	0.155
1	0.122	0.076	0.279	0.107	0.240
2	0.365	0.215	0.047	0.015	0.122
3	0.157	0.161	0.42	0.327	0.341
4	0.175	0.274	0.220	0.321	0.168



# Positional embeddings

Weisfeiler-Lehman Absolute Role Embedding

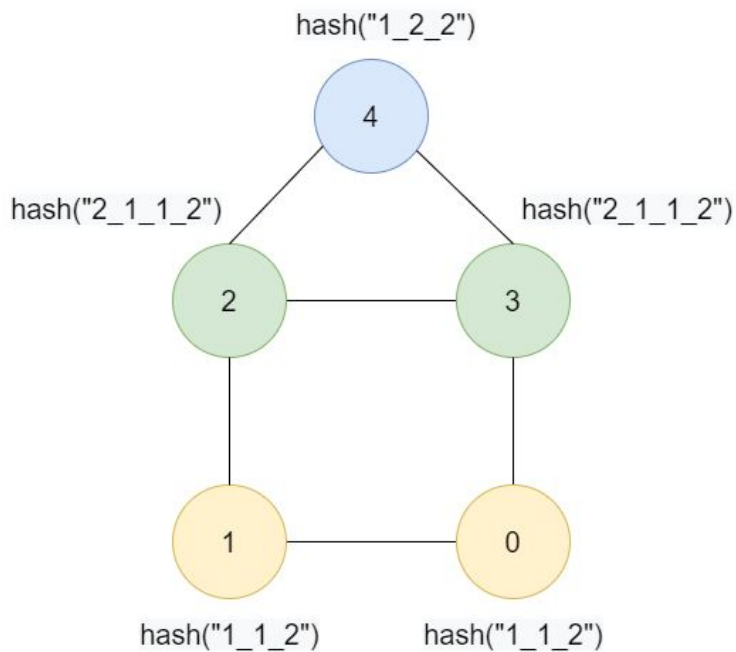
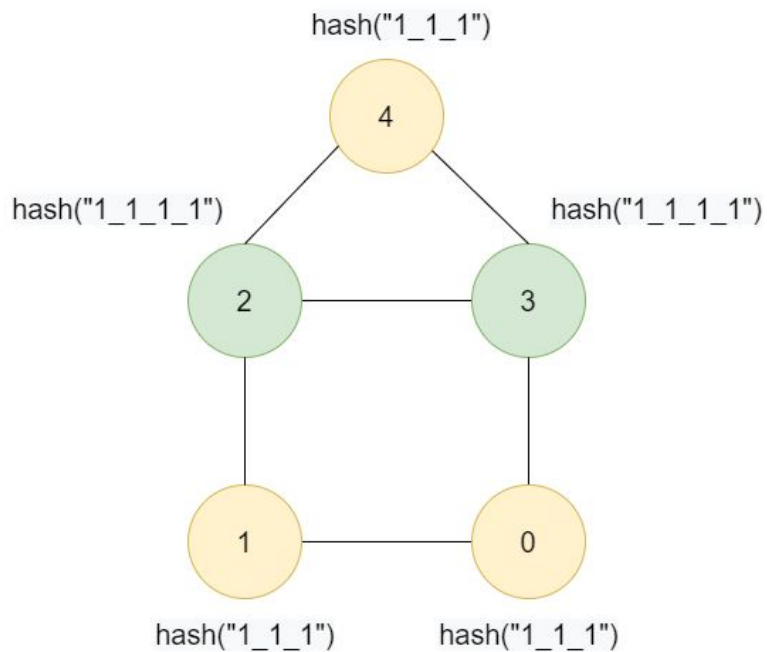
Intimacy based Relative Positional Embedding

Hop based Relative Distance Embedding



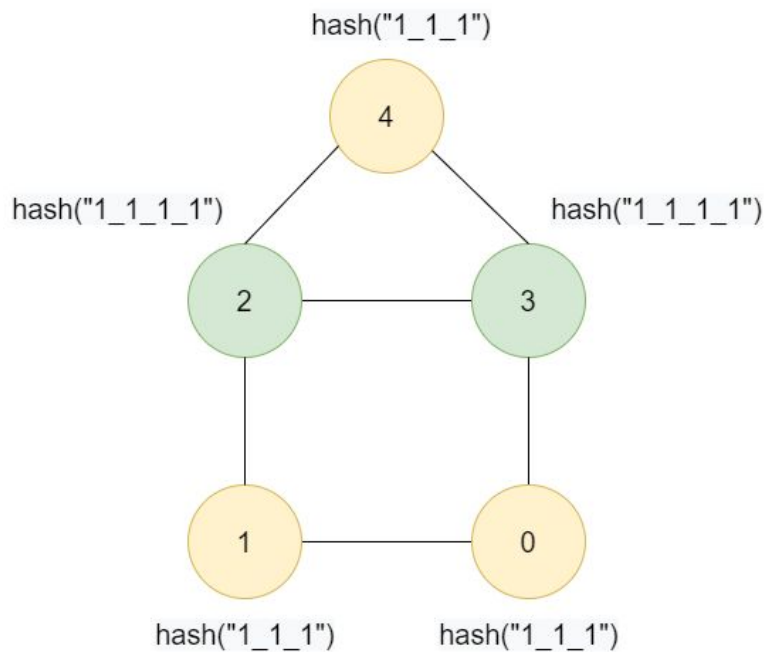
# Positional embeddings

## Weisfeiler-Lehman Absolute Role Embedding



# Positional embeddings

## Weisfeiler-Lehman Absolute Role Embedding



$$\begin{aligned} \mathbf{e}_j^{(r)} &= \text{Position-Embed}(\text{WL}(v_j)) \\ &= \left[ \sin\left(\frac{\text{WL}(v_j)}{10000^{\frac{2l}{d_h}}}\right), \cos\left(\frac{\text{WL}(v_j)}{10000^{\frac{2l+1}{d_h}}}\right) \right]_{l=0}^{\lfloor \frac{d_h}{2} \rfloor} \end{aligned}$$

# Positional embeddings

## Intimacy based Relative Positional Embedding

	0	1	2	3	4		target node	top k context
0	0.014	0.209	0.231	0.228	0.155	⇒	0	2 3 1
1	0.122	0.076	0.279	0.107	0.240	⇒	1	2 4 0
2	0.365	0.215	0.047	0.015	0.122	⇒	2	0 1 4
3	0.157	0.161	0.42	0.327	0.341	⇒	3	2 3 1
4	0.175	0.274	0.220	0.321	0.168	⇒	4	2 3 1
								0 1 2 3

$$\mathbf{e}_j^{(p)} = \text{Position-Embed}(\mathbf{P}(v_j)) \in \mathbb{R}^{d_h \times 1}$$

# Positional embeddings

Hop based Relative Distance Embedding

$$\mathbf{e}_j^{(d)} = \text{Position-Embed}(\text{H}(v_j; v_i)) \in \mathbb{R}^{d_h \times 1}$$

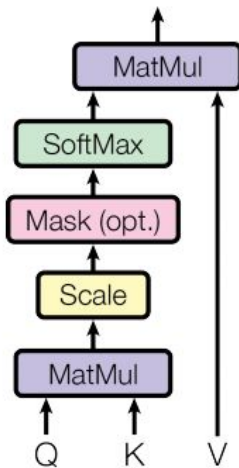
# Graph Transformer based Encoder

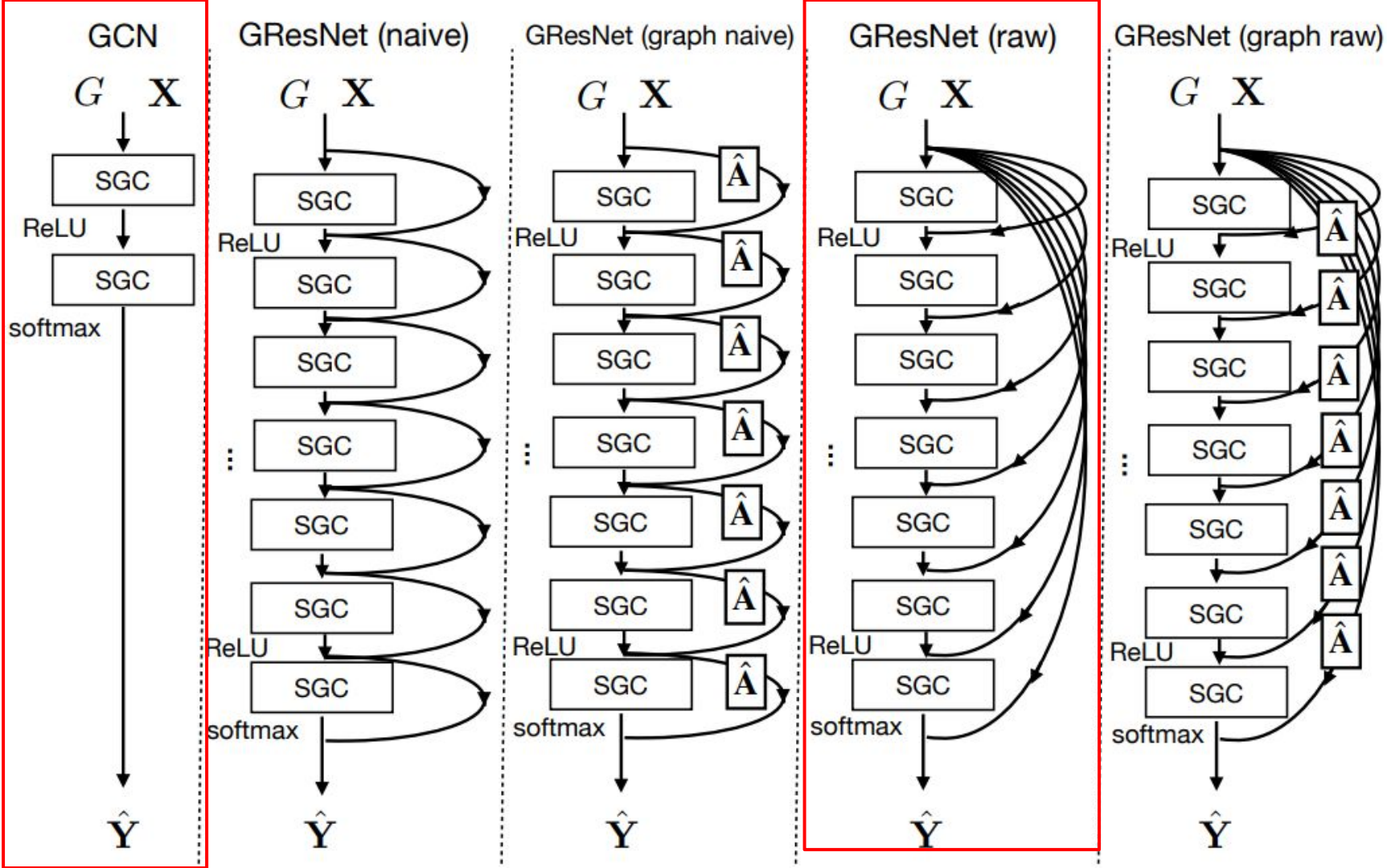
$$\begin{aligned}\mathbf{H}^{(l)} &= \text{G-Transformer}(\mathbf{H}^{(l-1)}) \\ &= \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_h}}\right)\mathbf{V} + \text{G-Res}(\mathbf{H}^{(l-1)}, \mathbf{X}_i)\end{aligned}$$

where

$$\begin{cases} \mathbf{Q} &= \mathbf{H}^{(l-1)}\mathbf{W}_Q^{(l)}, \\ \mathbf{K} &= \mathbf{H}^{(l-1)}\mathbf{W}_K^{(l)}, \\ \mathbf{V} &= \mathbf{H}^{(l-1)}\mathbf{W}_V^{(l)}. \end{cases}$$

Scaled Dot-Product Attention





# Graph Transformer based Encoder

$$\begin{cases} \mathbf{H}^{(0)} = [\mathbf{h}_i^{(0)}, \mathbf{h}_{i,1}^{(0)}, \dots, \mathbf{h}_{i,k}^{(0)}]^\top, \\ \mathbf{H}^{(l)} = \text{G-Transformer}(\mathbf{H}^{(l-1)}), \forall l \in \{1, 2, \dots, D\}, \\ \mathbf{z}_i = \text{Fusion}(\mathbf{H}^{(D)}). \end{cases}$$

# GRAPH-BERT Learning

Task #1: Node Raw Attribute Reconstruction

$$\ell_1 = \frac{1}{|\mathcal{V}|} \sum_{v_i \in \mathcal{V}} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2$$

$$\hat{\mathbf{x}}_i = \text{FC}(\mathbf{z}_i)$$

Task #2: Graph Structure Recovery

$$\ell_2 = \frac{1}{|\mathcal{V}|^2} \|\mathbf{S} - \hat{\mathbf{S}}\|_F^2$$

$$\hat{\mathbf{S}} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|} \text{ with entry } \hat{\mathbf{S}}(i, j) = \hat{s}_{i,j}$$



# Graph Transformer Networks

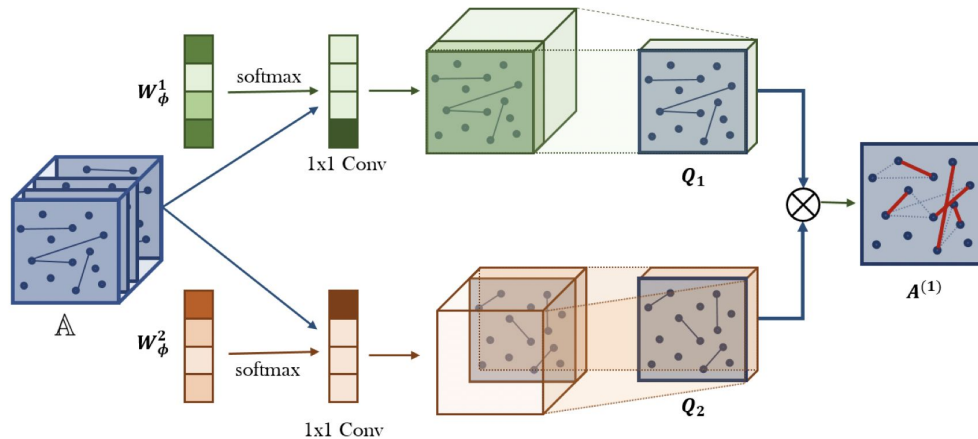


Figure 1: **Graph Transformer Layer** softly selects adjacency matrices (edge types) from the set of adjacency matrices  $\mathbb{A}$  of a heterogeneous graph  $G$  and learns a new meta-path graph represented by  $A^{(1)}$  via the matrix multiplication of two selected adjacency matrices  $Q_1$  and  $Q_2$ . The soft adjacency matrix selection is a weighted sum of candidate adjacency matrices obtained by  $1 \times 1$  convolution with non-negative weights from  $\text{softmax}(W_\phi^1)$ .

# Graph Transformer Networks

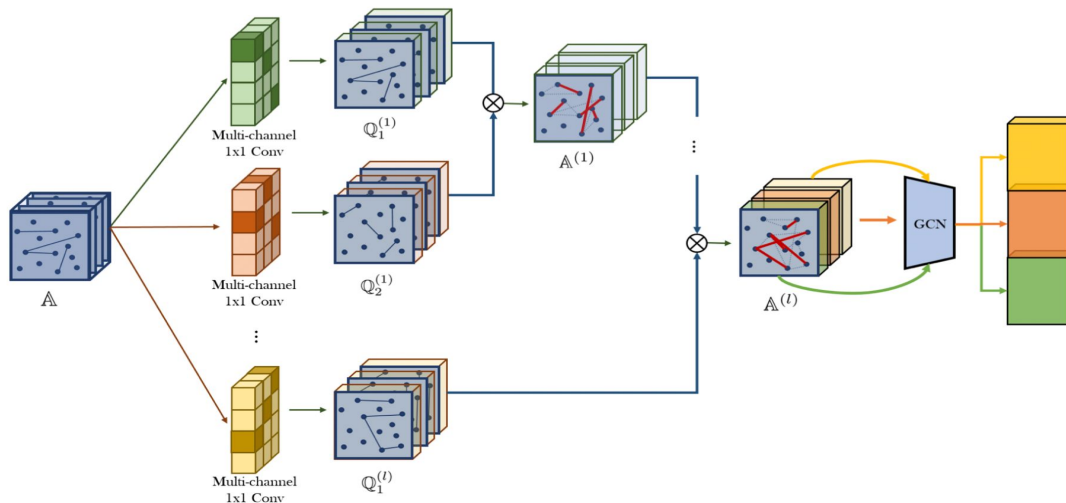
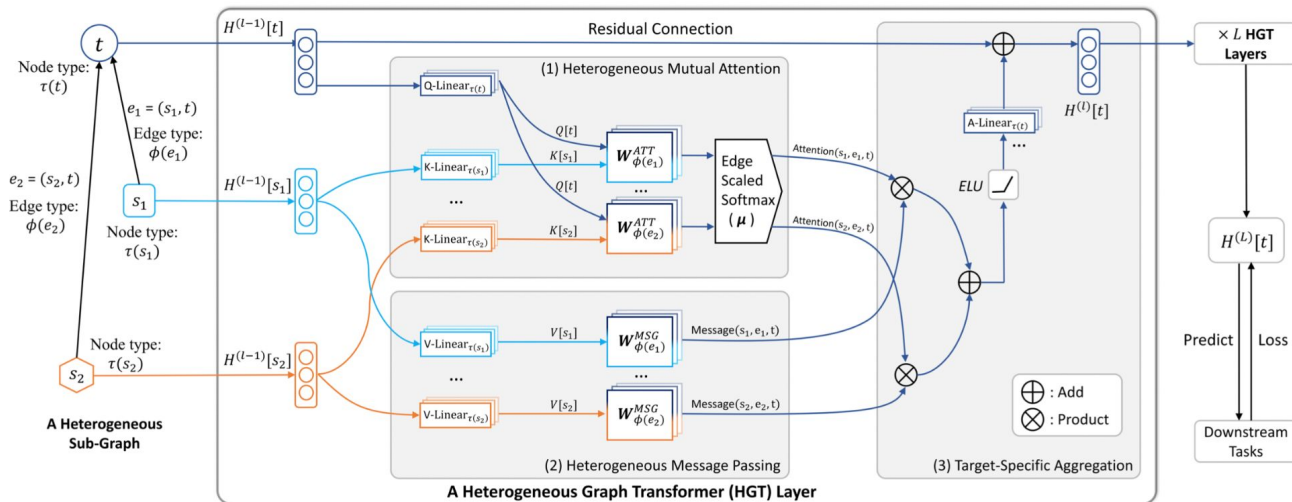


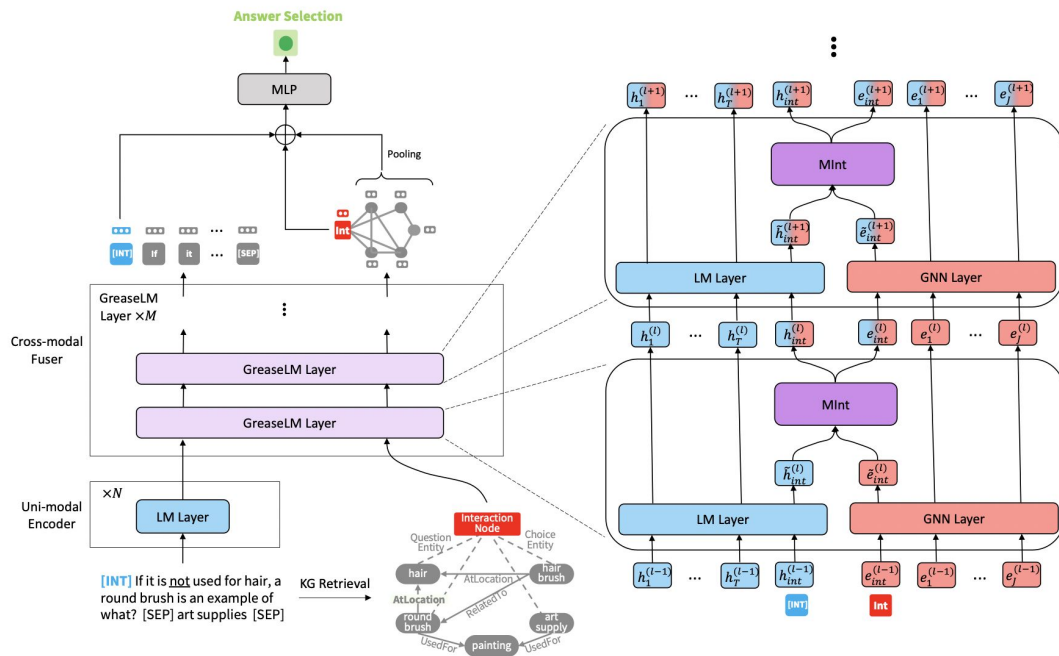
Figure 2: Graph Transformer Networks (GTNs) learn to generate a set of new meta-path adjacency matrices  $\mathbb{A}^{(l)}$  using GT layers and perform graph convolution as in GCNs on the new graph structures. Multiple node representations from the same GCNs on multiple meta-path graphs are integrated by concatenation and improve the performance of node classification.  $\mathbb{Q}_1^{(l)}$  and  $\mathbb{Q}_2^{(l)} \in \mathbf{R}^{N \times N \times C}$  are intermediate adjacency tensors to compute meta-paths at the  $l$ th layer.

# Heterogeneous Graph Transformer

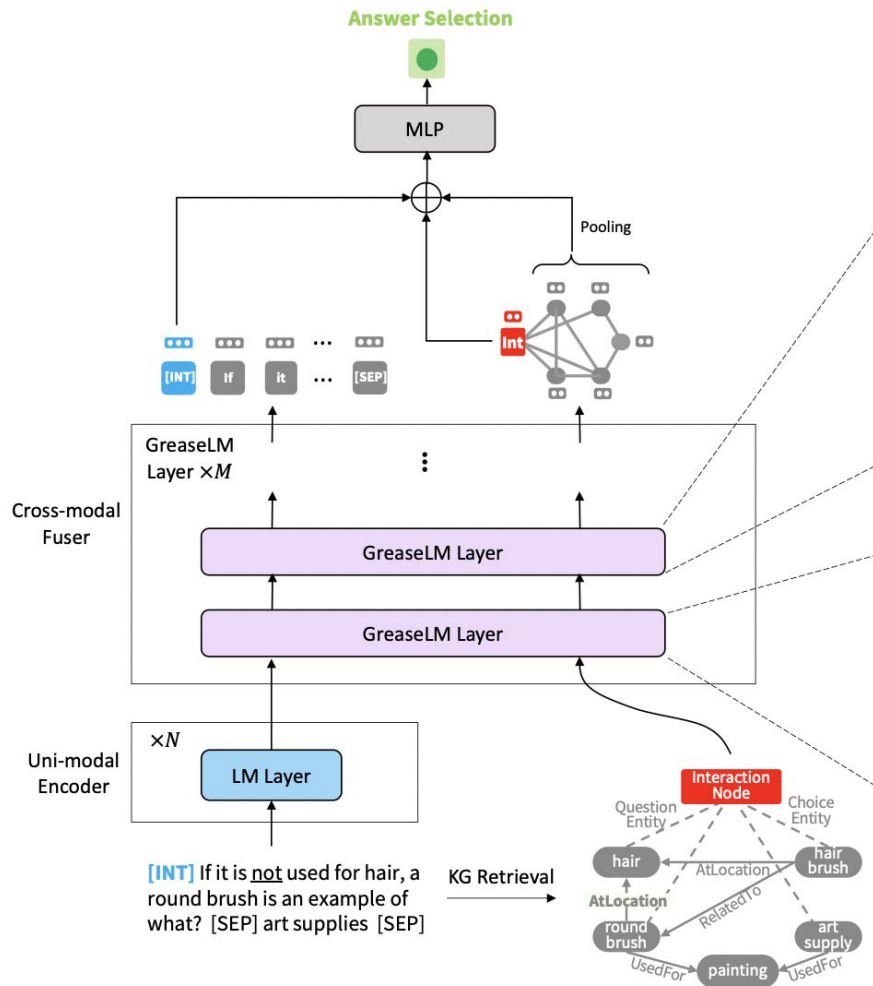


**Figure 2: The Overall Architecture of Heterogeneous Graph Transformer.** Given a sampled heterogeneous sub-graph with  $t$  as the target node,  $s_1$  &  $s_2$  as source nodes, the HGT model takes its edges  $e_1 = (s_1, t)$  &  $e_2 = (s_2, t)$  and their corresponding meta relations  $\langle \tau(s_1), \phi(e_1), \tau(t) \rangle$  &  $\langle \tau(s_2), \phi(e_2), \tau(t) \rangle$  as input to learn a contextualized representation  $H^{(L)}$  for each node, which can be used for downstream tasks. Color decodes the node type. HGT includes three components: (1) meta relation-aware heterogeneous mutual attention, (2) heterogeneous message passing from source nodes, and (3) target-specific heterogeneous message aggregation.

# GreaseLM: Graph REASoning Enhanced Language Models for Question Answering



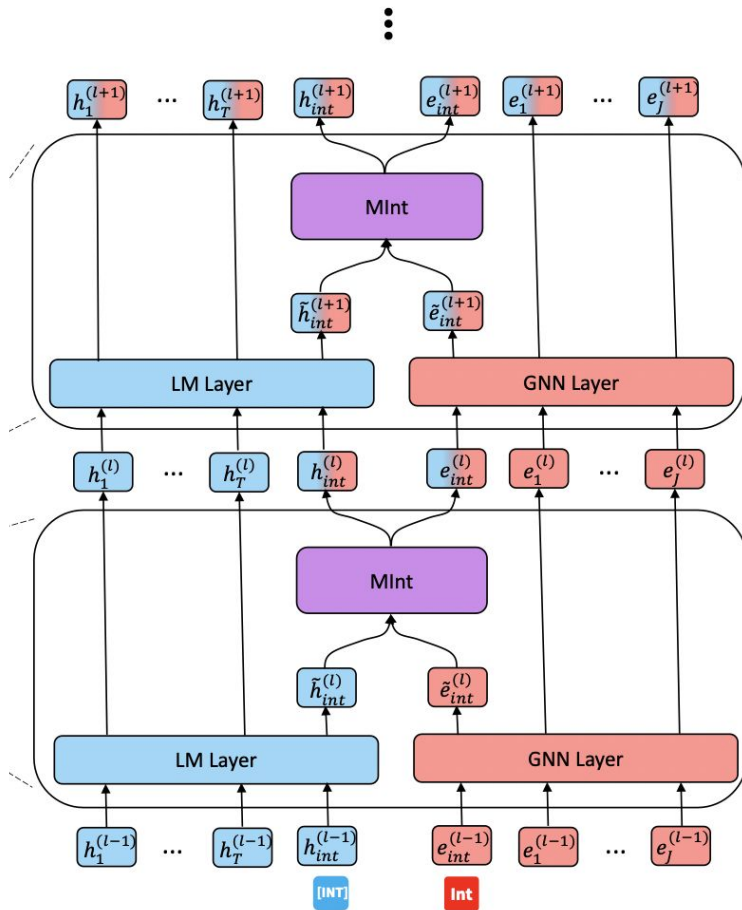
# GreaseLM



# GreaseLM

GNN corresponds to the **GAT** layer

LM model stands for  
**RoBERTa-Large / AristoRoBERTa  
/ SapBERT**



# Conclusion

- Transformers are also used for graph structures
- Many ways to present nodes and edges as input
- More graph transformers:

<https://github.com/ChandlerBang/awesome-graph-transformer>